DOCUMENT RESUME

ED 148 331 IR 005 387

AUTHOR Wescourt, Keith T.; And Others
TITLE Knowledge-Based Adaptive Curriculum Sequencing for CAI: Application of a Network Representation.
INSTITUTION Stanford Univ., Calif. Inst. for Mathematical Studies in Social Science.
SPONS AGENCY Navy Personnel Research and Development Center, San Diego, Calif.; Office of Naval Research, Washington, D.C. Psychological Sciences Div.
REPORT NO IMSSS-TR-288
PUB DATE 20 Sep 77
CONTRACT N00014-76-C-0165; N00123-75-C-1528
NOTE 22p.

EDRS PRICE MF-$0.83 HC-$1.67 Plus Postage.
DESCRIPTORS Artificial Intelligence; *Computer Assisted Instruction; *Concept Teaching; Curriculum Design; *Curriculum Development; *Problem Solving; Programing; *Sequential Approach; Technical Education
IDENTIFIERS BASIC Instructional Program; Curriculum Information Network

ABSTRACT

One aspect of tutoring skill for technical subjects is individualized, adaptive sequencing of the problems given to students as learning exercises. A Curriculum Information Network (CIN) describes the relationships between the problems in a CAI curriculum and the concepts and skills that they are intended to teach. It is a basis for selecting problems for each student with respect to his evolving knowledge of those concepts and skills. This paper describes the application of a semantic network to represent the complex interrelationships among the skills in a CIN for the BASIC Instructional Program, a CAI problem-solving laboratory for introductory programming in the BASIC language. The semantic network is used in drawing complex inferences about the student's state of knowledge and the problems that are appropriate to present to him. Such inferences enable more skillful problem sequencing by the CAI system. (Author)

****
* Documents acquired by ERIC include many informal unpublished *
* materials not available from other sources. ERIC makes every effort *
* to obtain the best copy available. Nevertheless, items of marginal *
* reproducibility are often encountered and this affects the quality *
* of the microfiche and hardcopy reproductions ERIC makes available *
* via the ERIC Document Reproduction Service (EDRS). EDRS is not *
* responsible for the quality of the original document. Reproductions *
* supplied by EDRS are the best that can be made from the original. *
****

U.S. DEPARTMENT OF HEALTH, EDUCATION & WELFARE
NATIONAL INSTITUTE OF EDUCATION

THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL NATIONAL INSTITUTE OF EDUCATION POSITION OR POLICY

ED148331

IR

IR005387

KNOWLEDGE-BASED ADAPTIVE CURRICULUM SEQUENCING FOR CAI: APPLICATION OF A NETWORK REPRESENTATION

by

Keith T. Wescourt, Marian Beard, and Laura Gould

TECHNICAL REPORT NO. 288

September 20, 1977

PSYCHOLOGY AND EDUCATION SERIES

To be published in the Proceedings of the National Association for Computing Machinery, 1977.

This research was supported by the Personnel and Training Research Programs, Psychological Sciences Division, Office of Naval Research under Contract No. N00014-76-C-0165, Contract Authority Identification Number, NR No. 154-381 and by the Navy Personnel Research & Development Center under Contract N00123-75-C-1528.

Approved for public release; distribution unlimited.

Reproduction in whole or in part is permitted
for any purpose of the U. S. Government.

INSTITUTE FOR MATHEMATICAL STUDIES IN THE SOCIAL SCIENCES

STANFORD UNIVERSITY
STANFORD, CALIFORNIA

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Knowledge-based Adaptive Curriculum Sequencing for CAI: Application of a Network Representation		5. TYPE OF REPORT & PERIOD COVERED Technical Report 6. PERFORMING ORG. REPORT NUMBER Technical Report No. 288
7. AUTHOR(s) Keith T. Wescourt, Marian Beard and Laura Gould		8. CONTRACT OR GRANT NUMBER(s) N00014-76-C-0165
9. PERFORMING ORGANIZATION NAME AND ADDRESS Institute for Mathematical Studies in the Social Sciences, Stanford University, Stanford, California 94305		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61153N RR 042-06; RR 042-06-01 NR 154-381
11. CONTROLLING OFFICE NAME AND ADDRESS Personnel & Training Research Programs Office of Naval Research (Code 458) Arlington, VA 22217		12. REPORT DATE September 20, 1977 13. NUMBER OF PAGES 7
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Aspects of this research were also supported by the Navy Personnel Research & Development Center under contract N00123-75-C-1528. This report will be published in the Proceedings of the National Association for Computing Machinery, 1977

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

computer-assisted instruction (CAI), instruction control strategy, optimized learning, computer programming, BASIC Instructional Program (BIP), semantic network, artificial intelligence (AI), student model, inference procedures, Curriculum Information Network (CIN)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

One aspect of tutoring skill for technical subjects is individualized, adaptive sequencing of the problems given to students as learning exercises. A Curriculum Information Network (CIN) describes the relationships between the problems in a CAI curriculum and the concepts and skills that they are intended to teach. It is a basis for selecting problems for each student with respect to his evolving knowledge of those concepts and skills. This paper describes the application of a semantic network to represent the complex

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102 LF 014 6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

interrelationships among the skills in a CIN for the BASIC Instructional Program, a CAI problem-solving laboratory for introductory programming in the BASIC language. The semantic network is used in drawing complex inferences about the student's state of knowledge and the problems that are appropriate to present to him. Such inferences ebable more skillful problem sequencing by the CAI system.

KNOWLEDGE-BASED ADAPTIVE CURRICULUM SEQUENCING FOR CAI: APPLICATION OF A NETWORK REPRESENTATION

Keith T. Wescourt, Marian Beard, and Laura Gould[1]

Institute for Mathematical Studies in the Social Sciences
Stanford University
Stanford, California

One aspect of tuturing skill for technical subjects is individualized, adaptive sequencing of the problems given to students as learning exercises. A Curriculum Information Network (CIN) describes the relationships between the problems in a CAI curriculum and the concepts and skills that they are intended to teach. It is a basis for selecting problems for each student with respect to his evolving knowledge of those concepts and skills. This paper describes the application of a semantic network to represent the complex interrelationships among the skills in a CIN for the BASIC Instructional Program, a CAI problem-solving laboratory for introductory programming in the BASIS language. The semantic network is used in drawing complex inferences about the student's state of knowledge and the problems that are appropriate to present to him. Such inferences enable more skillful problem sequencing by the CAI system.

1. Introduction

Since its inception a major goal of computer-assisted instruction (CAI) has been to individualize instruction by making the teaching system's behavior contingent on a student's responses during prior interactions with the system. Some early adaptive CAI systems applied features of mathematical learning and decision theories to sequencing drill-and-practice curricula in elementary mathematics (Suppes & Morningstar, 1972) and in initial reading (Groen & Atkinson, 1966). These systems generated problems of different types (e.g., addition problems

with or without "carries') according to an estimate of the student's mastery of each type based on his prior history of correct and incorrect answers. As a result, more capable students could progress rapidly to harder problem types and remediation and review could be determined with respect to individual difficulties manifested with specific problem types.

More recently, features adapted from Artificial Intelligence (AI) systems have been applied to individualize CAI along other dimensions. Research on question answering an natural language has enabled systems such as SCHOLAR (Collins, Warnock, & Passafiume, 1975) to conduct spontaneous instructional dialogues, giving students great freedom of expression and providing exposure to a topic according to the student's demonstrated familiarity with it. Other systems, for example, SOPHIE (Brown & Burton, 1975), have been designed to monitor student problem-solving attempts and react to errors or requests for help with comments that reflect the context in which the student has been working. Significant progress has been made in AI-based CAI toward understanding the student's behavior with respect to representations of the knowledge he acquires and uses, and not merely in terms of his performance on particular questions and problems.

Our research over the past few years has attempted to apply some AI techniques to individualized curriculum sequencing for complex scientific and technical subjects. Traditionally, instruction in these types of subjects has depended heavily on students solving large numbers of problems. This method has proven to be effective for forcing the integration of concepts and problem-solving skills described in lectures and readings. Our concern has been to individualize the sequencing of a curriculum of complex problems in order to improve the acquisition

and integration of the underlying concepts and skills of the subject. The result of our initial efforts was the concept of the Curriculum Information Network (CIN), a means of describing the relationships among the problems in a curriculum in terms of procedural skills involved in solving them, and for modeling student learning with respect to those same skills (Barr, Beard, & Atkinson, 1976). A CIN and a problem-selection procedure were implemented in the BASIC Instructional Program (BIP), a CAI system that teaches the BASIC programming language at an introductory level. The present paper describes our recent research, in which we developed a semantic network representation for structural and pedagogical knowledge about BASIC and some inference techniques for using this network to increase the sophistication of CIN-based problem selection in the BIP system.

2. Overview of the BIP system

2.1 Design

BIP is designed to teach elementary programming concepts and skills without lectures or a standard textbook. All of the resources available to students reside within the BIP system, illustrated in Figure 1, except for a manual written especially for the course. Furthermore, after a brief interactive introductory lesson. BIP presents no further text lessons on programming; instead, the student learns to program by solving programming problems using a BASIC interpreter built into the instructional system. For the purposes of our research, the value of BIP as a stand-alone CAI resource is that the effects of problem swquencing on student performance are emphasized and can be evaluated more directly than they could in a multifaceted system involving lectures, readings, and question-and-answer sessions in addition to a problem-solving laboratory.

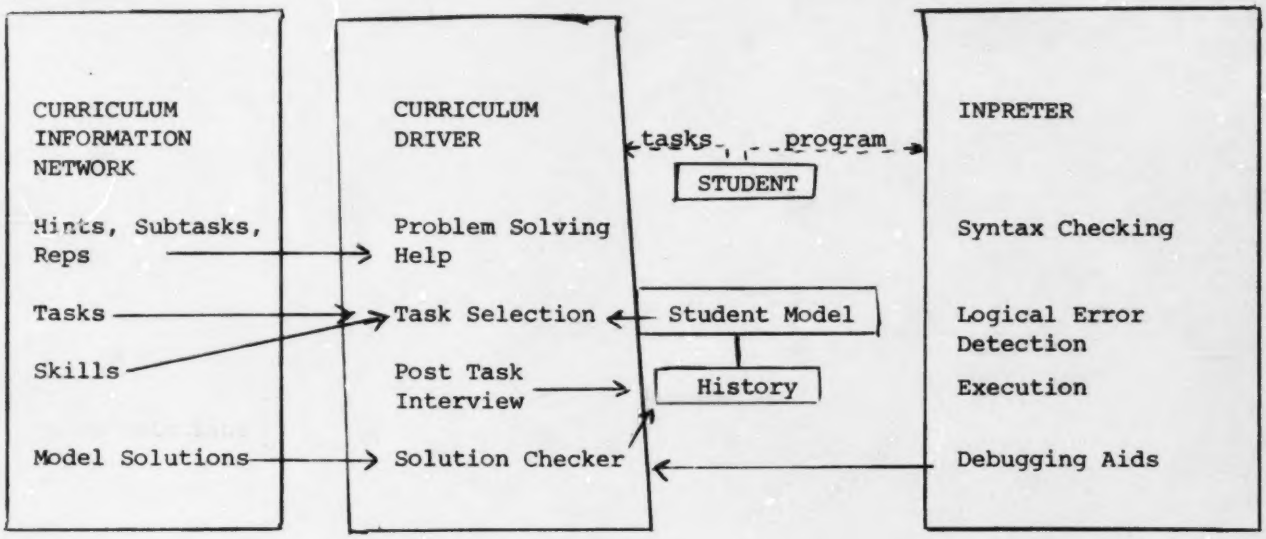

Figure 1. A schematic representation of the BIP CAI system.

2.2 Curriculum

The only parts of the BIP system that we need to consider in this paper are those involved in its curriculum-sequencing process. The curriculum consists of about 100 human-authored programming problems, which we refer to as tasks. While BIP's task-selection procedures could work as well, or better, if tasks were generated dynamically, at present there is no way to generate complex tasks that are as realistic and motivating as those written by a person. Furthermore, in a real instructional situation, it is the more usual case that a teacher starts with a set of tasks he feels are appropriate and then considers a task-sequencing strategy rather than vice-versa. The following is a typical problem for the BIP curriculum:

> On the first day of Christmas, someone's true love sent him/her a partridge in a pear tree (one gift on the first day). On the second day, the true love sent two turtle doves in addition to another partridge (three gifts on the second day). This continued through the 12th day, when the true love sent 12 lords, 11 ladies, 10 drummers, . . . all the way to yet another partridge.
>
> Write a program that computes and prints the number of gifts sent on that twelfth day. (This is not the same as the total number of gifts sent as the total number of gifts sent throughout all 12 days--just the number sent on that single 12th day.)

2.3 The CIN: Relating tasks and skills

The primary knowledge base in BIP is a Curriculum Information Network. In the CIN of BIP-I, the version described by Barr, et al. (1976), each task in the curriculum is described in terms of a set of procedural skills necessary to complete it successfully (i.e., to write the BASIC program that is called for). About 90 skills are used to describe the entire BIP curriculum, with some tasks involving as few as one or two skills, and others more than ten. Each student's progress through the course is modeled with respect to his learning of the total set of skills. The skills defined in the total set of skills. The skills defined in BIP-I in no way constitute a sufficient basis for representing the complete process by which a student understands a problem, determines a solution algorithm, and implements it as acceptable BASIC code. The skills relate only to the coding process, many corresponding to a single line of code. For example, if a problem solution includes the lines

```
. . .
70 INPUT T
80 PRINT R * T
. . .
```

then its description in the CIN would include the skills, "assign numeric variable by -INPUT-" and "print numeric expression [operation on variables]".

Although the skills are, for the most part, defined in terms of syntactic constructs, BIP-I task selection does not reflect the student's knowledge of syntax, but instead depends on his knowledge of the semantics and pragmatics for using the skill appropriately. All purely syntactic errors are detected immediately by the BIP-I interpreter, which responds with explicit feedback describing the error and illustrating syntactically correct examples of the construct. These syntactic

errors do not affect the model of the student maintained by BIP-I. This model is affected by logical errors which allow the student's program to run, but not to produce correct results. Many of these errors can be associated with semantically or pragmatically inappropriate use of the syntacti constructs described by BIP-I skills.

In BIP-I, the skills are grouped into about a dozen non-overlapping sets called techniques, such as simple printing, assignment, and conditional branching. The techniques themselves are linearly ordered according to judgments about the relative complexity of the skills they contain. The technique ordering is used in BIP-I as a constraint on the order in which major concepts are introduced and as a scale for determining whether problems available for remediating particular skills are appropriate to the student's overall progress.

2.4 Using a CIN for task selection

The general paradigm for applying a CIN to problem selection is:

1. From the model of the student's learning of all the skills, assemble sets of skills that correspond to relevant pedagogical considerations, such as "need further work," "ready to be learned," "already learned," etc.
2. Using these sets and other historical data (e.g., those tasks that this student has already completed), search the CIN for the task that uses a set of skills most congruent to a set that best satisfies overall pedagogical goals. For example, given the overall goal of reducing the total number of tasks students must complete to reach a level of competence, one might choose to search for a task with the most "need further work" and "ready to be learned" skills and the fewest "already learned" skills.

3. Present the task to the student and analyze his performance on it to update the model of his learning of the skills.

Three main theoretical entities bear on this task-selection paradigm. First, there is the representation of knowledge to be learned, which is the skills in the CIN. Second, there is a theory of learning, which maps performance on tasks onto changes in the student's knowledge of the skills. Finally, there is a theory of instruction, which for any state of knowledge determines the next task that is "best" for the student to work on.

3. Applying a network representation of knowledge to task selection

Our initial use of a CIN in BIP-I for selecting tasks has demonstrated the successful application of the CIN paradigm (Barr, et al., 1976). However, the representation of programming knowledge and the model of student learning used in BIP-I are very rudimentary. Most obviously, BIP-I's grouping of skills into techniques is an oversimplification of the actual interrelations between skills. The technique groups do not provide a sufficient basis for anticipating a student's performance in new contexts based on his performance in related contexts--an important aspect of a human tutor's skill in selecting tasks for his student. Likewise, the student model, consisting of counters for each skill, does not differentiate various levels of skill mastery indicated by the amount of difficulty a student encounters in completing tasks. We therefore undertook to design a new CIN-based task-selection procedure for a BIP-II system, incorporating both a more detailed representation for the knowledge underlying the curriculum and more complex assumptions for modeling student learning. The remainder of this paper will focus on the new representation we have developed and how it is used in the BIP-II system.

In considering alternative representations for the knowledge underlying a task, we recognized that the most powerful approach would be a procedural representation sufficient to synthesize task solutions (see, for example, Brown, Burton, Miller, deKleer, Purcell, Hausmann, & Bobrow, 1975, and Carr & Goldstein, 1977). However, the state-of-the-art in program synthesis and analysis techniques has not yet advanced to a point where a manageable system could be implemented for automatically solving programming problems like those in the BIP curriculum. Thus, we decided to extend the original concept of a set of skills by embedding the skills in a network representation describing the structural and pedagogically significant relations between them. The network relationships allow inferences that potentially add sophistication to both the process of task selection and of interpreting student performance to update the student model. For example, unlearned skills that are deemed to be analogous to other skills that are already learned can be given lower priority for inclusion in the next task to be presented. Or, if such skills occur in a task that a student quits, then they can be taken as less likely sources of his difficulty than unlearned skills that are analogous to other skills that are already known to be troublesome for that student.

3.1 The BASICNET

Rather than basing the network of knowledge to learned directly on the BIP curriculum, we built the skill relationships on a general representation for BASIC programs. From a general analysis of the BASIC language, guides to BASIC programming, and the skills and techniques of BIP-I, we developed a network representation for BASIC programming constructs (the BASICNET), a simplified portion of which is shown in Figure 2.

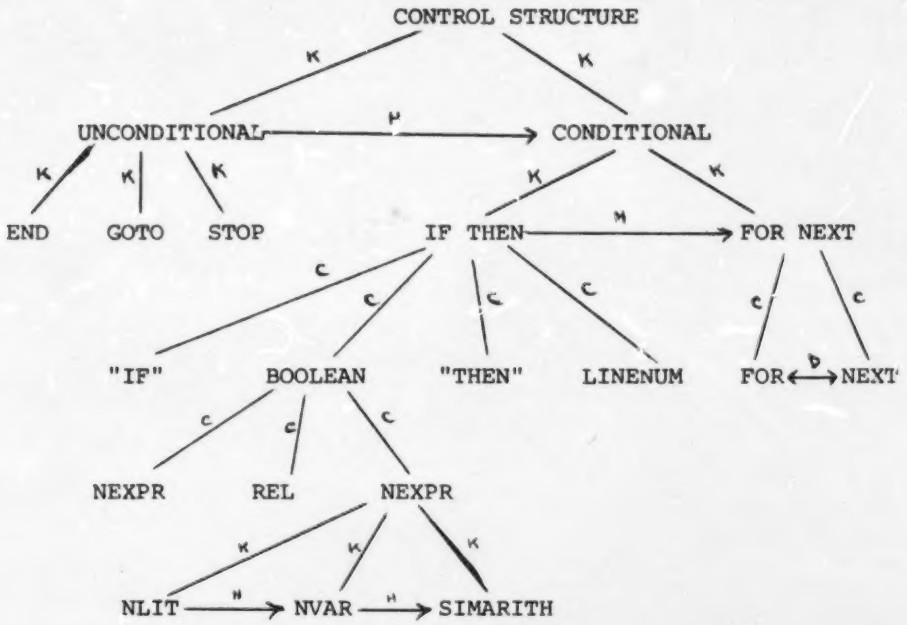

Figure 2. A simplified portion of the BASICNET describing control structures of the BASIC language.

The node names are self-explanatory; the links (relationships) are Kind, Component, Hardness, and (mutual functional) Dependency. The section of the BASICNET shown specifies that there are two kinds of control structures, and expresses a judgment that the conditional kind is harder to learn than the unconditional. There are two kinds of conditional structures, and FOR'NEXT is harder than IF'THEN. The components of an IF'THEN statement are the words "IF" and "THEN" with the Boolean condition and the line number in the appropriate places. For the purposes of this illustration, the BOOLEAN consists of a numeric expression (NEXPR), a relational operator (REL), and another NEXPR; among the three kinds of NEXPRs, numeric literals (NLIT) are easiest, and numeric variables (NVAR) and simple arithmetic expressions (SIMARITH) are increasingly hard.

Note that the downward links in the figure provide information like that found in a BNF notation for BASIC, while the horizontal links provide pedagogical information specifying relative difficulty, analogy, and dependency. The opinions expressed by the horizontal links are necessarily general and do not always hold for all students in all stages of learning. For example, an arithmetic expression is generally a harder construct that a numeric variable because it often includes a variable itself, but observation indicates that using a statement such as PRINT 6+4 tends to be an easier task for a beginner than using PRINT N. This implies that, ultimately, the pedagogical relationships between concepts must sometimes be a function of the student's state of learning at the time the relationships are to be used. We have chosen not to tackle this refinement in the BASICNET underlying the BIP-II System.

3.2 List notation for the BASICNET

A simplified version of the list natation we use to represent the protion of the BASICNET in FIGURE 2 is:

```
(CONTROL'STRUCTURE K (UNCONDITIONAL CONDITIONAL))
(UNCONDITIONAL K (END GOTO STOP) H (CONDITIONAL))
(CONDITIONAL K (IF'THEN FOR'NEXT))
(IF'THEN C ("IF" BOOLEAN "THEN" LINENUM) H
     (FOR'NEXT))
(FOR'NEXT C (FOR NEXT))
(FOR D (NEXT))
(BOOLEAN C (NEXPR REL NEXPR))
(NEXPR K (NLIT NVAR SIMARITH))
(NLIT H (NVAR) A (SLIT))
(NVAR H (SIMARITH) A (SVAR) S (SVAR))
```

The A links specify that numeric literals are analogous to string literals (SLIT), and that numeric variables are analogous to string variables (SVAR). The S link says that NVAR and SVAR are similarly difficult. (The information about SLIT and SVAR is found in another part of the ASICNET.) The notation here is simply that used to express property lists in LISP.

3.3 The BASICNET and BIP skills

After the BASICNET was defined, each skill in BIP's CIN was represented in terms of a subnet. First, the structure of each skill was described, in list notation like the following (where Skill 42 is "conditional branch, comparing a numeric literal with a numeric variable"):

```
(SK042 (IF'THEN (BOOLEAN . (NEXPR . NLIT)
                            (NEXPR . NVAR)))
```

Skill 42 is represented as an instance of IF'THEN (see Figure 2), in which the BOOLEAN component is further specified as consisting of the relation between a numeric literal (the first NEXPR component of the BOOLEAN) and a numeric variable (the second NEXPR). The REL is left uninstantiated, since Skill 42 does not specify the kind of comparison to be made between the two. Thus, any REL is appropriate.

Skill 43 is "conditional brance, comparing a simple numeric expression with a numeric variable." Its structure is

```
(SK043 (SK042 (NEXPR . SIMARITH)))
```

The notation is read "Skill 43 is identical to Skill 42 except that the first instance of NEXPR should be SIMARITH," which is exactly what the English description of the skill says. Skill 46 ("conditional branch, comparing two numeric variables") is represented as

```
(SK046 (SK042 (NEXPR . NVAR)))
```

again reflecting the minimal difference between the related skills.

3.5 The SKILLSNET

Within each skill set, pairs of skills were examined to find their minimal difference. If the nodes by which they differ are linked in the BASICNET, that link was used to define a relation between the skills. If the nodes by which two skills differ do not have a direct link, relations were sought at increasingly higher levels of the BASICNET.

For instance, since the BASICNET shows NVAR to be harder than 46. The relationships determined in this manner between all pairs of skills comprise the SKILLSNET, a knowledge representation that can be directly expressed in a CIN and used for task selection. The SKILLSNET, like the BASICNET, can be expressed in LISP property list notation. (The underlining in the following example emphasizes the relationships being discussed here.)

```
(SK042 H (SK044 SK046 SK047) A (SK047)
       P (SK003 SK036 SK039))
(SK043 H (SK047 SK075 SK061)
       P (SK036 SK040 SK003 SK005))
(SK046 H (SK043 SK048 SK045) A (SK048)
       P (SK003 SK036 SK039))
```

The P links shown here are Prerequisite links; like the Hardness links, these are a matter of pedagogical opinion. The P links, however, appear only in the SKILLSNET, not in the BASICNET, and express judgments that are more specific to the BIP course than those expressed in the BASICNET. A few of the skills (e.g., those involving the use of built-in BASIC functions such as INT and SQR) did not fall into skill sets since they seemed not to be describable in terms of any other skill. These are related within the SKILLSNET only by means of P links.

3.6. BIP-II task-selection procedure

The new task-selection procedure for the BIP-II system, designed to use the relationships between skills expressed in the SKILLSNET, is identical to the technique-based method in its overall design: A set of skills appropriate to the student's current level of understanding is generated, a set of tasks using some of those skills is identified, the best of those tasks (by some criteria) is presented, and the student model is updated based on the student's performance and self-evaluation on the task. The major difference between the two methods is that by using an expanded set of relations between skills in the SKILLSNET,

the new procedure can make more intelligent inferences both in updating the student model and in generating the set of skills to be involved in the student's next task.

The following is a simplified description of the precess by which a task is selected at any point during instruction, given the student's state of knowledge of all the skills. The procedure integrates a number of a priori reasonable pedagogical heuristics about how to vary the relative difficulty of tasks to optimize learning as performance varies and about how to teach a network of knowledge (e.g., breadth-first vs. depth-first exposure.)

STEP 1: Create a set called NEED, consisting of skills that will be sought in the next task. Look for "trouble" skills first (those in tasks that the student quit), then for analogies to learned skills, then for inverse-prerequisites of learned skills. As soon as a group of such skills is found, stop looking.

STEP 2: Remove from the NEED set those skills that have unlearned prerequisites. Add those skills to the NOTREADY set (which may be used later).

STEP 3: Given a NEED set, find the most appropriate task that involves some of the NEEDed skills.

(a) Assemble GOODLIST, those tasks that have the desired number of NEEDed skills. (This number increases if the student is consistently successful, descreases if he has trouble.)

(b) If no GOODLIST can be created, make a new NEED set consisting of the prerequisites of the skills in NOTREADY. If no new NEED set can be created, then the curriculum has been exhausted; otherwise, GOTO 3a.

(c) Find the "best" task: if the student is doing well, find the task in GOODLIST that has the fewest learned skills; if he is progressing more slowly, find the task with the fewest unseen skills. Remove the selected task from GOODLIST.

STEP 4: See if the selected task is otherwise appropriate.

(a) If none of the skills in the selected task have unlearned prerequisites, stop liiking and present the task. (END)

(b) If any skills have unsatisfied prerequisites, reject the task and add those skills to the NOTREADY set.

(c) If GOODLIST is exhausted, change (usually reduce) the criterial number of NEED skills, and GOTO 3a. Otherwise, using the rest of GOODLIST GOTO 3c.

As an example of the inferences made in the generation of the NEED set, let us assume that Skills 42 and 61 ("FOR . NEXT loops with literal as final value of index") are under consideration. Skill 61 is represented as

(SK061 H (SK042))

The Prerequitite relationship specified that 42 must be learned before a task involving 61 can be presented. The structure enforced by the P links relating pairs of skills gives the presentation of tasks some degree of order, and is designed to prevent too-rapid progress or drastic jumps in difficulty. The Hardness links, in contrast, are used to facilitate progress for a student doing well, by allowing some skills to be considered "too easy" for inclusion in the NEED set. Such skills are not inferred to be learned; they are simply not sought actively by the selection algorithm.

As an example using the skills described here, suppose that a student has successfully completed a task using Skill 43, although he has not yet seen Skill 42. (The fact that 43 is harder than 42 does not force 42 to be presented first; only P links force such order.) When the task-selection procedure assembles the next set of NEED skills, it will "infer" that 42 is now too easy to become part of that set, since something harder that 42 has already been learned.

Furthermore, since 42 is now considered too easy to liik for, Skill 61 can now be sought. If the student successfully completes a task involving 61, the student model will be updated to show that 61 has been learned, and by inference, that its unseen prerequisite 42 has also been learned. These kinds of inferences (by which skills can reach too-easy or learned states without the student actually having

seen them in a task) can of course be contradicted by direct observation or by other inferences if the student has difficulty. For example, the unseen prerequisite of a given skill may change its state from "too easy" to "in trouble" if the student quits (gives up on) a task involving the given skill. The next task selection would attempt to find a task using that prerequisite skill in such a case.

3.7 BIP-II performance

The BIP-II task-selection procedure has been implemented with parameters (e.g., numbers of skills sought and thresholds determining when an unpresented skill is "too easy" to be included in the NEED set) that can be changed readily. The system can therefore be used to explore the effectiveness of somewhat different pedagogical heuristics for task sequencing. We used this capability in conjunction with a simulation system we developed to create a version of BIP-II that we expected would be effective for a range of student abilities. Recently, we collected data from a group of about 27 students who used this BIP-II system. At this time, the data have not been extensively analyzed, but we can provide a general summary and report our subjective observations.

The students were limited to fifteen hours of terminal time with BIP. They were presented with (but did not necessarily complete successfully) an average of 40 tasks, the minimum being 21 tasks and the maximum 63 tasks. Only one student dropped out of the course without completing fifteen hours or finishing the curriculum.

One overall measure of the success of semantic network CIN and related task-selection procedure is the relationship between number of skills learned (according to BIP) and scores on a paper-and-pencil posttest. The correlation between these measures was .86, and accounts for 74 percent of the variability in the posttest scores. The students also took a standardized test of programming aptitude prior to instruc-

tion. The pretest scores correlated significantly with both the number of skills learned (r = .65) and the posttest scores (r = .59); however, in a multiple regression of the posttest scores on the number of skills learned and the pretest scores, only the number of skills learned contributed significantly to posttest performance. Number of tasks presented to students was independent of both test scores and number of skills learned. Thus, the student model maintained by the BIP-II system accurately reflects the acquisition of the programming knowledge required by the posttest, and predicts posttest performance independently of the aptitude measured by the pretest.

Our informal observations indicate that BIP-II task sequences are substantially different from those of BIP-I. Most noticeably, when a student does well initially, BIP-II selects conplex tasks at a much earlier point than they were selected by BIP-I. Students' reactions were favorable, even when they spent considerable time on these difficult tasks and then gave up. Task selection following these failures seems responsive: BIP-II selected simpler tasks involving some "not-learned" skills that were involved in the task that the student had quit. In many cases, these "remedial" tasks appear to have been too simple, given the student's prior progress, but most often BIP-II was in fact looking for a more challenging task and could not find one in the curriculum.

Besides the sequencing "failures" due to the limits of the curriculum, a large number of inadequate task-selection decisions were caused by poor data from the system's solution checker. In these cases, the student had a substantially correct program that was rejected by the checker. BIP-II interprets the rejections as a sign that the student is having difficulty with some of the skills in the task and thereby

introduced errors into the student model. Sometimes students in our study became so frustrated by the rejection of their programs that they quit the task, creating even more severe errors in the model.

Disregarding the difficulties caused by these weaknesses elsewhere in the instructional system, our initial evaluation of BIP-II's task selection capabilities is favorable. However, it is clear that evaluation of the effects of manipulating the parameters of the selection process will be impeded until the curriculum is expanded and the solution checker is improved.

4. Summary and conclusions

We have indicated how complex knowledge representations adapted from AI research can be used to describe a CAI problem curriculum and thereby enable relatively sophisticated individualized problem sequencing. In particular, a CIN based on a semantic-network representation provides a medium for drawing indirect inferences about what a student knows, what he is ready to learn, and what task in the curriculum will best help him learn it. A network representation then is useful not only for expressing unambiguous relationships (e.g., property-inheritance, componency), its typical application in AI systems, but, in addition, is one way for systematically expressing opinions about the pedagogical relationships among the concepts and skills a CAI system is intended to teach.

We do not believe that the BIP-II system, based on a CIN incorporating a complex network of skill relationships, can match a human tutor's ability to select programming problems adaptively: The limitations imposed by the system's rudimentary program checker insure some extreme failures, but, beyond this, the SKILLSNET and the inferences that use it only approximate the flexibility of which a tutor is capable. Nonetheless, the more evolved CIN makes it possible for tutorial CAI in technical subjects to individualize student experience effectively across a range of student

abilities and instructional goals.

Notes

This research was supported by the Office of Naval Research, Personnel & Training Research Programs, under contract number N00014-76-C-0165, and by the Naval Personnel Research & Development Center under Contract N00123-75-C-1528.

[1] Laura Gould is now at Xerox Palo Alto Research Center.

References

Barr, A., Beard, M., & Atkinson, R. C. The computer as a tutorial laboratory: The Stanford BIP project. International Journal of Man-Machine Studies, 1976, 8, 567-596.

Brown, J. S., & Burton, R. R. Multiple representation of knowledge for tutorial reasoning. In D. G. Bobrow and A. M. Collins (Eds.), Representation and understanding: Studies in cognitive science. New York: Academic Press, 1975.

Brown, J. S., Burton, R., Miller, M., deKleer, J., Purcell, S., Hausmann, C., & Bobrow, R. Steps toward a theoretical foundation for complex, knowledge-based CAI. BBN Report No. 3]35, Bolt Beranek and Newman, Inc., Cambridge, Mass. August, 1975.

Carr, B., & Goldstein, I. P. overlaps: A theory of modelling for computer aided instruction. MIT AI Memo 406, Massachusetts Institute of Technology, Artificial Intelligence Laboratory, Cambridge, Mass. February, 1977.

Collins, A. M., Warnock, E. H., & Passafiume, J. J. Analysis and synthesis of tutorial dialogs. In C. H. Bower (Ed.), The psychology of learning and motivation (Vol. 9). New York: Academic Press, 1975.

Groen, G. J., & Atkinson, R. C. Models for optimizing the learning process. Psychological Bulletin, 1966, 4, 309-320.

Suppes, P., & Morningstar, M. Computer-assisted instruction at Stanford, 1966-68: Data, models, and evaluation of the arithmetic programs. New York: Academic Press, 1972.